多人数不完全情報ゲームに対する局面評価値を用いたモンテカルロ法 (計算理論とアルゴリズムの新潮流)

AUTHOR(S):

漆畑, 雅士

CITATION:

漆畑, 雅士. 多人数不完全情報ゲームに対する局面評価値を用いたモンテカルロ法 (計算理論とアルゴリズムの新潮流). 数理解析研究所講究録 2014, 1894: 84-88

ISSUE DATE:

2014-05

URL:

http://hdl.handle.net/2433/195834

数理解析研究所講究録
第1894巻 2014年 84-88

多人数不完全情報ゲームに対する局面評価値を用いたモンテカルロ法

中央大学大学院理工学研究科 漆畑雅士
Masashi Urushibata
Faculty of Science and Engineering, Chuo University

1 序論

近年，モンテカルロ法により コンピュータ囲碁のAIは急激に強くなっている．囲碁は，将棋やチェスのようなボードゲームとは違い，合法手をランダムに打ち合っても終局させられるという点においてモンテカルロ法に都合がよかったためである．モンテカルロ法はさまざまなゲームAIの研究において注目されている．その中で，多人数不完全情報ゲームであるトランプゲーム“大貧民”においても，プレイヤープログラムの強さを競うUECコンピュータ大貧民大会 (UECda) が毎年開催され，モンテカルロ法を用いたプレイヤープログラムが活躍している [1]．本研究では，大貧民を題材とし，モンテカルロ法に利用されるUCB1に局面評価値を実装することによって，精度の向上を図るとともにより強いプログラムを構築することを目指す．

2 大貧民

大貧民はトランプを用いて遊ぶゲームの一つで，多人数不完全情報ゲームに分類される．トランプを全参加者に配り，それぞれが手持ちのカードを順番に場に出して早く手札をなくすことを競うゲームである．1試合のみで終了することはほとんどなく，複数回試合を行うことが多い．大きな特徴として，1試合終了時の順位が次試合開始時の有利不利に影響するという点が挙げられ，勝者が有利な状況から次の試合を始める．本研究でのルールはUEC標準ルール [1] に則っている．

3 モンテカルロ法

3.1 大貧民におけるモンテカルロ法

以下で説明するアルゴリズムは「各候補手に対して，プレイアウトを何回も行い，勝率の最も高い候補手を選択する」というモンテカルロ法をもとにしている．プレイアウトとは，乱数を用いてゲームを終局までプレイすることをいう．大貧民におけるモンテカルロ法は以下のような手順で行う．

1. 合法手の列挙
   自身が選択可能な合法手を列挙する．
2. プレイアウトを指定回数行う．
   ここで，合法手の選択，手札の配布，乱数によるシミュレーション，報酬値のフィードバックを行う．報酬値として，プレイアウトの結果が大富豪ならば5，富豪は4，平民は3，貧民は2，大貧民は1を返す．これを指定回数繰り返す．
3. 報酬値の比較
   全プレイアウト終了後，各合法手に対する報酬値の平均を比較し，それが最大の合法手を最善手として出力する．

3.1.1 UCB1-TUNED

多腕バンディット問題は, 複数のマシンを多数回プレイする中で, 一番利益を得るには, どのマシンをプレイするのがいいか, ということを決定する問題である. この問題に対して, UCB1(Upper Confidence Bounds) というアルゴリズムが提案されている [2]. これは, 結果が悪かったマシンで施行回数が多いものはプレイせず, 逆に施行回数が少ないマシンはプレイするアルゴリズムである. 具体的には UCB1 値によってマシンの選択を行う. ここで, 総プレイ回数を n としたとき, その n 回のうち, マシン i がプレイされた回数を $T_i(n)$, $\overline{X}_i$ をマシン i が $T_i(n)$ 回プレイされた時の利益の平均とすると, UCB1 値は

$$\overline{X}_i + \sqrt{\frac{2\ln n}{T_i(n)}}$$

によって表される. UCB1 アルゴリズムは上記の値が最も高いマシン i に対してプレイアウトを行う. 更に, この UCB1 アルゴリズムにおいて, 各マシンの経験値の分散を考慮したものが UCB1-TUNED[2] であり, 具体的には

$$\overline{X}_i + \sqrt{\frac{V \times \ln n}{T_i(n)}}$$

$$V = \min(0.25, \overline{X_i^2} - (\overline{X_i})^2 + \sqrt{\frac{2\ln n}{T_i(n)}})$$

を用いるものである. 本研究では, UCB1-TUNED を利用して, 勝率の高い候補手に, より多くのプレイアウトを割り当てる. ここで, 前述のマシンは候補手に対応し, 利益は報酬値に対応する. 大貧民における報酬値は, 結果が大富豪ならば 5, 富豪ならば 4,..., 大貧民なら 1 の値を取る.

3.2 差分学習法の応用

モンテカルロ法は, プレイアウト途中における盤面を評価せずに報酬値のフィードバックを行っている. そこで, 途中経過も含めて強化学習を行うことにより, モンテカルロ法よりも適切な行動を選択する方法が提案されている [3]. 例えば, 差分学習法に基づいて報酬値を変化させたものが

$$V = \sum_{t=0}^{N-1} r_t \omega_t$$

である. すなわち, プレイアウト中に生成された盤面状態 t において, 盤面の評価値 r_t を算出する. そしてその盤面における評価の重み ω_t に従って最初に選択した合法手 i に逐次的なフィードバックを行う.

3.3 局面評価値の導入

UCB1-TUNED は, 最初に選択する合法手の評価を全て均一のものとしてシミュレーションを行う. そこで, UCB1-TUNED に局面評価値を導入することで局面評価値が良いと思われる手を優先して探索する. この手法は, 中華圏で人気のゲーム "Big Two" において性能向上を示し, その有効性が示されている [4]. 実際に UCB1-TUNED に局面評価値を導入したものが

$$\overline{X}_i + \sqrt{\frac{V \times \ln n + C_E \times E_i}{T_i(n)}}$$

$$V = \min(0.25, \overline{X_i^2} - (\overline{X_i})^2 + \sqrt{\frac{2\ln n}{T_i(n)}})$$

である. ただし C_E は定数であり, 計算機実験の項で詳述する. また, E_i は局面 i における局面評価値である. 上式より明らかに, $n_i \to \infty$ となるにつれて勝率 $\overline{X_i}$ が重視されるようになる.

局面評価値の作成に当たって, "Big Two" のゲーム AI, "coffee AI" を参考にした [5]. 実際に coffee AI を参考に作成した局面評価値 E_i は

$$E_i = E_{sp} + \frac{E_{sz}}{1000} + \frac{1000 - E_{sum}}{1000^2} + E_p$$

である. ただし各記号は以下の通りである.

E_{sp} : 残り手札を全て提出するための最小手番数

E_{sz} : 残り手札の枚数

E_{sum} : 残り手札中のカードの強さの総和.

E_p : 強いカードを使っているか, 次の手番でカード

を出せるか，よく多くの枚数を使うカードの組合せを崩していないか，パスを選択しているか，しばり手かどうかをそれぞれを考慮した評価値．

coffee AI の評価関数を利用するに当たり，カードの強さの調整，Big Two には存在しないルールであるしばり手を考慮した評価値，そしてより強い手を崩さないようにする評価値を追加した．導入に際し，局面評価値は 0.0〜1.0 の値を取るように調整を行った．

3.4 手札交換における手札分割

ゲーム開始前に大富豪，富豪は，大貧民，貧民にカードを渡す必要がある．原口は，提出することが可能な手に応じて手札を分割する手法を提案した [6]．例えば図 1 のような手札分割を行うと，手数が 4 の組合せと，手数が 3 の組合せができる．階段，トリプル，8 を切札とすると，それぞれの切札の枚数は 2 枚となる．手数ー切札の数 が 1 となった時，手札提出における読み切りが可能となるため，良いとされるものは右の組合せとなる．このようにして，手数ー切札の数 がより小さい分割を，良い分割と考える．実際の運用は切札数ではなく，各手の切札度の総和で行う．この手札分割を手札交換において利用する．最適な組合せを見つけた後に，ソロのカードの中で最も弱いカードを交換する．最小値探索には分枝限定法を用いる．

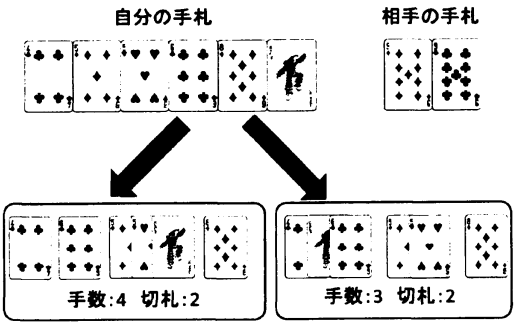

図 1: 手札分割の例

4 計算機実験

モンテカルロ法のプレイアウトにおいて UCB1-TUNED を利用するクライアントを “UCB1-TUNED.” UCB1-TUNED に局面評価値を導入したクライアントを “UCB+.” 差分学習法を応用したモンテカルロ法を UCB1-TUNED に導入したクライアントを “TD.” TD に局面評価値を導入したクライアントを “UCB+TD.” UCB+TD に手札分割を実装したクライアントを “UCB+TDhand” と呼ぶ．それぞれを用いた実験をして比較する．実験で記述する 1 ゲームは 1000 試合とする．

・計算機実験 1

計算機実験 1 では，UCB+と UCB1-TUNED の強さを比較する．UCB+1 つ，UCB1-TUNED1 つ，そして過去に UECda にて優勝したクライアント 3 つの計 5 つを戦わせる．1 ゲーム終了後，UCB+と UCB1-TUNED のポイント合計を比較し，点数の高い方を勝ちとして結果を出す．UCB+の局面評価値における C_E の値は 0.2〜2.2 で変化させ，それぞれ 40 ゲーム行う．プレイアウト数は UCB+，UCB1-TUNED 共に 2000 回とした．図 2 は UCB+の勝利数と敗北数を定数 C_E 別にまとめている．値

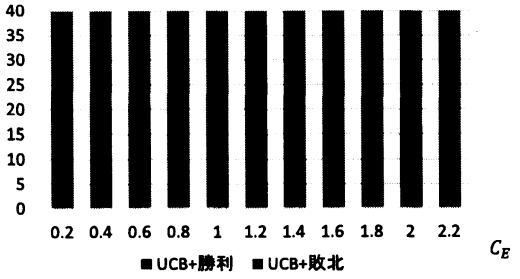

図 2: UCB+と UCB1-TUNED の対戦結果

$C_E = 1.2$ の時，UCB1-TUNED に対して 25 勝 15 敗という最も良い結果が得られたため，この定数を使って計算機実験 2 以降の実験を行っている．また，C_E が 1.6 より大きくなると，敗北数が勝利数を上回ってしまう．すなわち局面評価値への依存度が高過ぎると性能が下がってしまうことが分かる．

・計算機実験 2

計算機実験 2 では，UCB+TD と TD の強さを比較する．UCB+TD1 つ，TD1 つ，UCB1-TUNED3 つの計 5 つを戦わせる．1 ゲーム終了後，UCB+と UCB1-TUNED のポイント合計を比較し，点数の高い方を勝ちとして，100 ゲーム行う．プレイアウト数は全てのクライアントにおいて 2000 回とした．図 3 に，それぞれのクライアントの勝利数をまとめる．結果は UCB+TD の 61 勝 39 敗で，UCB+に対

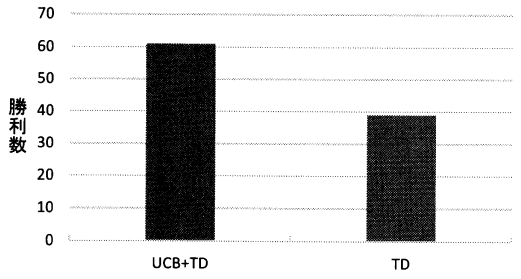

図 3: UCB+TD と TD の対戦結果

して 6 割程度の勝率となった．計算機実験 1 と合わせて，局面評価値を UCB1-TUNED に導入することで，精度が向上している．

・計算機実験 3

計算機実験 3 では，UCB+TDhand と UCB+TD の強さを比較する．UCB+TDhand1 つと UCB+TD4 つの計 5 つを同時に戦わせて比較を行う．プレイアウト数は 2000 回とし，100 ゲーム行う．図 4 は UCB+TD の 100 ゲームの結果である．UCB+TDhand は，大富豪 39 回，富豪 24 回，平民 16 回，貧民 14 回，大貧民 7 回という結果を出した．手札交換のプログラムが作用するのは，大富豪，富豪の時のみではあるが，それでも良い結果を残している．従来の手札交換プログラムは，手札を探索して単に弱いものをカード交換していたが，その交換によって手札を弱くしてしまっていると思われる．

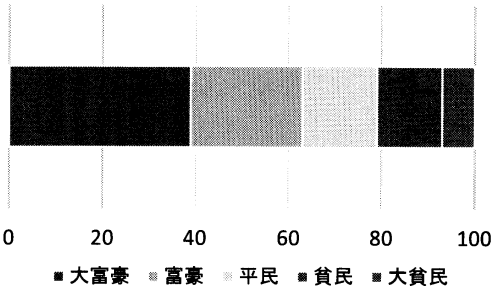

図 4: UCB+TDhand と UCB+TD の対戦結果

5 結論

多人数不完全情報ゲームである大貧民に対し，局面評価値を UCB1 値に導入することによって，精度の向上を図り，より強いプログラムを作成することができた．今回作成した局面評価値は，各プレイヤーの手札の状況のみを見ているだけで，盤面の状態を見てはいないが，それでも一定の効果が得られた．より強い局面評価値が作成できれば，性能を更に向上させることも可能であると考えられる．加えて，手札交換のプログラムを実装するだけで，強くなったことから，手札提出のみでなく，手札交換においても今後は注目していく必要があるだろう．

参考文献

[1] 電気通信大学，“UEC コンピュータ大貧民大会” (available at http://uecda.nishino-lab.jp/).

[2] P. Auer, N. Cesa-Bianchi, P. Fischer, “Finite-time analysis of the multiarmed bandit problem,” *Machine Learning*, **47** (2002), pp. 235–256.

[3] 小沼 啓，本多 武尊，保木 邦仁，西野 哲朗，“コンピュータ大貧民に対する差分学習法の応用,” **情報処理学会ゲーム情報学研究会** *(SIGGI)* **研究報告**, **1** (2012), pp. 1–4.

[4] 万代 悠作, 橋本 剛, "UCB+を用いた Big Two AI の研究," ゲームプログラミングワークショップ *2012* 論文集, **6** (2012), pp.205–210.

[5] SourceForge.net: bigtwo, "Project Web Hosting - Open Source Software" (available at http://bigtwo.sourceforge.net/).

[6] 原口和也, 私信, 2013.